Videojuegos

Curso de Diseño y Programación

Nº 19 5,99 euros

*Implementando los **menús***

***Texturas** para los bonos*

*Efectos de **titulación** y **croma***

MULTIPLAYER SETUP

PLAYER SETUP

19







Curso práctico de Videojuegos

AUTOR DE LA OBRA
Marcos Medina
DIRECCIÓN EDITORIAL
Eduardo Toribio
etoribio@iberprensa.com
COORDINACIÓN EDITORIAL
Eva-Margarita García
eva@iberprensa.com
DISEÑO Y MAQUETACIÓN
Antonio Gª Tomé
PRODUCCIÓN
Marisa Cogorro
SUSCRIPCIONES
Tel: 91 628 02 03
Fax: 91 628 09 35
suscripciones@iberprensa.com
FILMACIÓN: Fotpreim Duvial
IMPRESIÓN: Gráficas Don Bosco
DUPLICACIÓN CD.ROM: M.P.O.
DISTRIBUCIÓN
S.G.E.L.
Avda. Valdelaparra 29 (Pol. Ind.)
28108 Alcobendas (Madrid)
Tel.: 91 657 69 00

EDITA: Iberprensa
www.iberprensa.com



CONSEJERO
Carlos Peropadre

REDACCIÓN, PUBLICIDAD Y ADMINISTRACIÓN
C/ del Río Ter, 7 (Pol. Ind. "El Nogal")
28110 Algete (Madrid)
Tel.: 91 628 02 03
Fax: 91 628 09 35
(Añada 34 si llama desde fuera de España.)

DEPÓSITO LEGAL: M-35934-2002
ISBN: **Coleccionable:** 84 932417 2 5
Tomo 2: 84 932417 4 1
Obra Completa: 84 932417 5 X

NOTA IMPORTANTE:
Algunos programas incluidos en los CD de "Programación y Diseño de Videojuegos" son versiones completas, pero en otros casos se trata de versiones demo o trial, versiones de evaluación que Iberprensa quiere ofrecer a nuestros lectores. No se trata en ningún caso de las versiones comerciales de los programas, y las hemos incluido para dar al lector la oportunidad de conocer y probar esos programas y que así pueda decidir posteriormente si desea o no adquirir las versiones comerciales de cada uno.

Aprende divirtiéndote

Bienvenidos de nuevo a **Programación y Diseño de Videojuegos**, la primera obra coleccionable cuyo objetivo es formar al alumno en las principales técnicas relacionadas en el desarrollo completo de un videojuego.

A lo largo de la obra el lector está aprendiendo programación a nivel general y a nivel específico con ciertas herramientas y lenguajes, aprendiendo a trabajar con aplicaciones de retoque de imagen y también de diseño 3D y animación. Estamos descubriendo las aplicaciones profesionales más importantes de audio y conociendo la historia de lo que se denomina "la industria del videojuego", los últimos 20 años, los juegos que marcaron un avance, sus creadores y en general la evolución del videojuego.

Pero además, esta obra tiene un segundo objetivo, desarrollar y potenciar la creatividad del lector, nosotros a lo largo de las diferentes entregas pondremos las bases y tú pondrás tu ingenio, tu creatividad y tu capacidad de mejorar.

Nos encontramos a mitad de camino del viaje de 20 semanas que os proponemos, viaje articulado en 400 páginas y 20 CD-ROMs cuya finalidad es proporcionar las bases mínimas para después cada uno continuar su camino.

Recuerda que para alcanzar el éxito necesitas cumplir tres condiciones: que te gusten los juegos, poseer cierta dosis de creatividad y finalmente capacidad de estudio.

Una la cumples seguro.

sumario



PARA ENCUADERNAR LA OBRA:

- *Tapas del volumen 1 disponibles en Iberprensa. Pedidos por teléfono: 916280203.*
- ***Tapas del volumen 2 ya a la venta en quioscos.***
- ***Los suscriptores recibirán las tapas en su domicilio sin cargo alguno como obsequio de Iberprensa.***

SERVICIO TÉCNICO:
Para consultas, dudas técnicas y reclamaciones Iberprensa ofrece la siguiente dirección de correo electrónico: games@iberprensa.com

PETICIÓN DE NÚMEROS ATRASADOS:
El envío de números sueltos o atrasados se realizará contra reembolso del precio de venta al público más el coste de los gastos de envío. Pueden ser solicitados en el teléfono de atención al cliente 91 628 02 03

Los menús del juego

En esta entrega desarrollaremos el código necesario para implementar los menús del juego.

El diseño de los menús de un juego difiere según sus características estéticas, pero básicamente, todos tienen unas opciones básicas para controlar el juego, los controles y los aspectos gráficos y sonoros.

(■) ANÁLISIS ESTÉTICO

En "Zone of Fighters" disponemos de todas estas opciones básicas alojadas en submenús que parten de un menú principal. En las figuras 1 y 2 se muestra un esquema de esta clasificación. Estéticamente, la idea básica radica en visualizar las diferentes opciones de los menús bien alineadas en el centro de la pantalla (independientemente de su resolución) sobre un fondo móvil que representa el terreno de combate. Se mantiene, de esta forma, un aspecto gráfico sencillo con una estética sobria. Al paso del puntero del ratón (representado por un punto de mira) sobre cada una de las opciones provoca que éstas cambien de color. El cambio de un grupo de opciones a otro se realiza de forma instantánea, sustituyendo una lista por otra (Fig. 3).

NUEVO JUEGO
Generada
Zona editada
Modo entrenamiento
Multijugador

NIVEL DE DETALLE
Detalle del terreno
Sombreado del terreno
Suavizado de bordes
Rango de visualización

1

Esquema de los submenús del juego (I).

Cada opción de una lista esta descrita por una frase (ejemplo: "OPCIONES") con el tipo de letra especialmente preparada para el juego (sección "gráficos" número 6) creada previamente como imagen bitmap. Por lo tanto, para cada opción disponemos de dos imágenes diferentes; una con la frase de color blanco y otra de color amarillo, la cual servirá de sustituto al paso del puntero del ratón. Estas imágenes las denominaremos "botones" para identificarlas en el código.

(■) ANÁLISIS PREVIO DEL CÓDIGO

Todo el código del menú está implementado en el módulo "Menu.bb" y consta de 4 funciones: "Menu()", "Panel_Opciones()","Tabla_Records" y "Fondo_Menu".

La función "Menu()" es donde inicializamos el menú y donde alojamos el bucle principal. Desde este bucle llamamos a las diferentes opciones que implementamos en la función "Panel_Opciones()". La función "Tabla_Records()" simplemente visualiza la tabla de puntuación cada cierto tiempo de inactividad. Y por último, la función "Fondo_Menu()" se encarga de visualizar el terreno de combate que sirve de fondo del menú.

(■) FUNCIÓN "MENU()"

Cada vez que el programa entra en esta función se cargan los ficheros de imágenes que participan en el mismo y cada vez que se sale se borran. De esta forma independizamos más esta función del resto del juego con lo que podemos diseñar el menú lo



Esquema de los submenús del juego (y II).

completo que queramos disponiendo de toda la capacidad del sistema. Dentro de la función, el primer paso es inicializar las variables de los parámetros de opciones cuando el juego se ejecuta por primera vez. Aunque perfectamente este procedimiento también se podía haber colocado en el módulo de definiciones. Seguidamente, cargamos todas las imágenes para cada botón (opciones), cursor y cada uno de los paneles. Antes de seguir, vamos a detallar la base del sistema utilizado para implementar la elección de cada opción.

En todo el menú y submenús disponemos de 45 botones dife-



El paso a los submenús se realiza inmediatamente, sustituyendo un grupo de opciones por otro al hacer clic.

Eleccion_Panel(2)
DETALLE DEL TERRENO
BAJO MEDIO ALTO
Boton(7) Boton(8) Boton(9)

4

Esquema del uso de la matriz "Eleccion_Panel(16)" y la matriz de botones "Boton(44)".

rentes que representan un total de 17 elecciones; es decir, si elegimos la opción "NIVEL DE DETALLE" encontramos cuatro opciones diferentes representadas por distintos botones. Por ejemplo, en la elección "Rango de Visualización" encontramos tres botones diferentes para representar los tres tipos de valores "BAJO MEDIO ALTO". Por lo tanto, para una elección tenemos tres botones distintos. Si sumamos todos los menús, encontramos 17 elecciones diferentes que suman un total de 45 botones.

Para un manejo más sencillo de cada elección disponemos de una matriz que contiene los tipos de elecciones diferentes que colocamos en el módulo de definiciones:

```
Dim Eleccion_Panel (16)
```

Para los botones definimos una estructura que contendrá el gráfico y las coordenadas de cada uno de ellos. Como tenemos 45 diferentes, creamos una matriz que contenga 45 de estas estructuras:

```
Type botones
     Field grafico
     Field x,y
End Type
Dim boton.botones(44)
Dim botonB.botones(44)
```

"boton.botones(44)" se utiliza para los botones de color amarillo que aparecen cuando el puntero del ratón se coloca sobre ellos y "botonB.botones(44) " para los botones de color blanco estándar.

De vuelta a la función "Menu()", después de cargar los archivos de imágenes, pasamos a crear cada uno de los botones y a asignarles el gráfico correspondiente. Como ejemplo, tomemos el proceso para algunos de ellos en el primero de los bucles:

```
For n=0 To 11
  boton(n)= New botones
  boton(n)\grafico=LoadImage
  ("c:\zone of fighters\gfx\
  menu\boton"+Str(n)+".png")
  botonB(n)= New botones
  botonB(n)\grafico=LoadImage
  ("c:\zone of fighters\gfx\
  menu\boton"+Str(n)+"B.png")
Next
```

Como podemos observar, a cada botón de la matriz se le asigna una imagen, cuyo archivo ha sido nombrado para que corresponda con el nuevo botón; es decir, para "boton(2)" se le asigna la imagen "boton2.png". El resto de creaciones y asignaciones funciona de la misma forma.

Antes de entrar en el bucle principal preparamos el fondo del menú. Para ello, creamos y posicionamos un terreno y colocamos una cámara que luego hará una panorámica continua de 360 grados.

Dentro del bucle principal, el primer paso es mantener la música de fondo sonando. Luego, llamamos a la función "Fondo_Menu()" a actualizar el movimiento de rotación de la cámara que muestra el terreno de fondo. Lo siguiente es imprimir la imagen del logo del juego en una esquina de la pantalla y todos los botones de color blanco correspondiente a las opciones principales:

```
DrawImage logotipo,20,
GraphicsHeight()-100
For n=0 To 6
  DrawImage botonB(n)\grafico,
  botonB(n)\x,botonB(n)\y
Next
```

Dim Eleccion_Panel(16)

2 - Detalle del terreno
3 - Sombreado del terreno
4 - Suavizado de bordes
5 - Rango de visualización
6 - Niebla
7 - Cielo
8 - Reflejo del agua
9 - Día y Noche

5

Lista de las primeras 8 opciones equivalentes a los valores 2 al 9 de la matriz "Eleccion_Panel(16)".

Para cambiar el color de las opciones al paso del puntero del ratón solo tenemos que preguntar si la imagen de éste colisiona con cualquiera de los botones. En caso afirmativo visualizamos el botón de color amarillo en el lugar del de color blanco. Además, chequeamos si durante la colisión se ha pulsado el botón derecho del ratón. Si es así, guardamos en la variable "eleccion" el número del botón correspondiente proporcionado por la variable del bucle "n". Posteriormente, el uso de este número nos servirá para dirigir el programa a una opción u otra:

```
For n=0 To 6
  If ImagesOverlap(cursor,
  MouseX(),MouseY(),botonB(n)\
  grafico,botonB(n)\x,botonB(n)\y)
    DrawImage boton(n)\grafico,
    botonB(n)\x,botonB(n)\y
    If MouseDown(1) eleccion=n
    Else eleccion=-1
  EndIf
Next
```

Como comentábamos, el valor contenido en "eleccion" lo utilizamos en una estructura "Select..Case" para dirigir el flujo del programa. En cada una de las opciones llamamos a la función "Panel _opciones()". Esta función es común para todas las opciones, así que tenemos que pasarle los parámetros necesarios para actúe de una u otra forma dependiendo de la elección (tipo de elección, panel gráfico, gráfico

Dim Eleccion_Panel(16)

10 - Densidad de decorado
11 - Decorado interactivo
12 - Terreno dinámico
13 - Volumen de la música
14 - Volumen efectos audio
15 - Nieve
16 - Lluvia

6

Lista de las últimas 7 opciones equivalentes a los valores 10 al 16 de la matriz "Eleccion_Panel(16)".

del puntero del ratón y gráfico del logotipo del juego):

```
Select eleccion
  Case 0:
  Panel_opciones(0,titulo(0),
  cursor,logotipo)
  Case 1:
    Panel_opciones(1,titulo(1),
    cursor,logotipo)
  ...
  ...
End Select
eleccion=-1
```

Para evitar mantener la última elección activa pasamos un valor de -1 a la variable "eleccion".

Para finalizar el bucle principal, dibujamos el puntero del ratón y controlamos la aparición de la tabla de puntuación dependiendo del valor del contador "aparece_tabla".

Salimos del bucle principal si pulsamos la tecla de escape o bien si hemos elegido la opción de jugar *(Play = True).*

No hay que olvidar que debemos borrar todas las imágenes cargadas al principio de la función.

(■) FUNCIÓN "PANEL_ OPCIONES()"

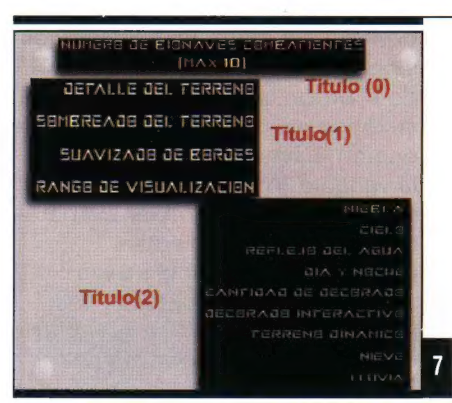
En esta función es donde implementamos el resto de submenús del juego. Como sabemos, accedemos a ella desde el bucle principal de la función "Menu()" con una serie de parámetros referentes al submenú que debemos visualizar. Estos parámetros son:

- **Tipo_panel:** corresponde al submenú de opciones seleccionado.
- **Grafico_panel:** corresponde al gráfico que representa la lista de opciones de cada submenú almacenada en la matriz "titulo(7)".
- **Cursor_raton** y **logo:** corresponden a los gráficos del puntero y del logotipo del juego que se muestra en la esquina de la pantalla.

Básicamente, esta función se basa en una estructura "Select.. Case" para ir a cada una de las opciones dentro de un bucle principal "Repeat.. Until", el cual termina cuando la variable "aceptar" valga 1; es decir, al pulsar el botón "OK".

Antes de entrar en la estructura "Select..Case" mantenemos la música del menú en continua reproducción, dibujamos el fondo del menú y el logotipo en la esquina.

La primera de las opciones se refiere al comienzo de una nueva partida en donde elegimos el tipo de juego que queremos emprender. El funcionamiento de esta elección fue comentada en el número 9 de esta sección cuando hablábamos de la elección del tipo de terreno de combate. Así que pasaremos por alto esta explicación para dejar paso al trabajo de la función en el resto de opciones.



Imágenes 0,1 y 2 correspondientes a los diferentes títulos de la matriz "Titulo(7)".

Prácticamente, el proceso que se sigue en las siguientes opciones es el mismo, por lo que comentaremos solo una de ellas. Elegimos pues la opción primera en la que modificamos el nivel de detalle gráfico. Lo primero que realizamos es que el programa visualice el gráfico del panel correspondiente contenido en la variable "grafico_panel". En este caso se trata de la imagen "titulo1.png" de la matriz "titulo(1)". Continuamos, visualizando todos los botones que corresponden a cada una de las opciones por cada elección (Ver Código 1).

Observamos que el primer botón que mostramos en todas las opciones es "botonB(38)", el cual corresponde a la opción "OK" en amarillo. Los siguientes botones ("botonB(7)", "botonB(8)", "botonB(9)", etc.) corresponden a las opciones "BAJO", "MEDIO" y "ALTO" para cada elección (Fig. 4).

Seguidamente, entramos en un bucle "For.. Next" para recorrer cada uno de los botones para preguntar si han colisionado con el

Código 1. Visualizamos todos los botones

```
Case 1
    DrawImage grafico_panel,((GraphicsWidth()/2)-(ImageWidth(grafico_panel)/2))-150,20
    botonB(38)\x=(GraphicsWidth()/2)+130: botonB(38)\y=180
    botonB(7)\x=(GraphicsWidth()/2)-(ImageWidth(grafico_panel)/2)+120:  botonB(7)\y=20
    botonB(8)\x=(GraphicsWidth()/2)-(ImageWidth(grafico_panel)/2)+180:  botonB(8)\y=21
    botonB(9)\x=(GraphicsWidth()/2)-(ImageWidth(grafico_panel)/2)+240:  botonB(9)\y=22
    ...
    ...
```

puntero del ratón. En caso de colisión preguntamos si se ha pulsado el botón izquierdo del ratón. Si se pulsa, es que se ha elegido una opción. Ésta estará contenida en la variable "n" del bucle, la cual nos servirá para cambiar los parámetros correspondientes.
Después de asignar el nuevo valor, se lo comunicamos a la matriz de elecciones para indicarle el nuevo cambio. Recordemos que disponemos de 17 elecciones diferentes, las cuales utilizaremos para decirle al programa qué botón debe estar en amarillo (seleccionado) o en blanco (deseleccionado). Por ejemplo, si elegimos la opción de nivel de detalle del terreno bajo, hemos seleccionado la opción 7 ("boton(7)", detalle del terreno) y el tipo de elección 2 (bajo). En las figuras 5 y 6 se muestran una relación con los tipos de elecciones diferentes contenidas en la matriz "eleccion_panel(16)" (Ver Código 2).

Por último, imprimimos todas las opciones elegidas por medio de los botones de color amarillo. Para ello, recorremos con un bucle "For.. Next" cada uno de los diferentes tipos de elección para esta opción e imprimimos todas las que tengan un valor distinto de -1; es decir, las que han sido seleccionadas previamente pasándoles el valor de "n":

```
For n=2 To 5
  b=eleccion_panel(n)
  If b<>-1 DrawImage boton(b)\
  grafico,botonB(b)\x,botonB(b)\y
Next
```

Antes de cerrar el bucle principal de esta función debemos chequear la pulsación del botón "OK" ("botón(18)") en todos los submenús con el procedimiento habitual (Ver Código 3).

(■) FUNCIÓN "TABLA_RECORDS()" Y "FONDO_MENU()"

La función "Tabla_Records()" simplemente carga el fichero de puntuación y lo visualiza sobre un panel gráfico sobre el fondo del menú durante un tiempo determinado contenido en la variable contador "contador". Por último, la función "Fondo_Menu()" actualiza la posición de la luz ambiente y mantiene la rotación de la cámara.

Como hemos observado, el sistema utilizado para implementar el menú del juego permite añadir de forma sencilla nuevas opciones, simplemente aumentando la capacidad de las matrices de elección y de los gráficos de botones y paneles.

En el próximo número...

... finalizaremos el desarrollo de "Zone of Fighters" implementando un sistema simple de multijugador utilizando el protocolo UDP.

Código 2. Tipos de elecciones diferentes

```
For n=7 To 16
  DrawImage botonB(n)\grafico,botonB(n)\x,botonB(n)\y
  If ImagesOverlap(cursor_raton,MouseX(),MouseY(),botonB(n)\grafico,botonB(n)\x,botonB(n)\y)
    DrawImage boton(n)\grafico,botonB(n)\x,botonB(n)\y
    If MouseDown(1)
      Select n
        Case 7: detalle_terreno=1000: eleccion_panel(2)=n
        Case 8: detalle_terreno=1500: eleccion_panel(2)=n
        Case 9: detalle_terreno=2500: Num_particulas=200:eleccion_panel(2)=n
        ...
        ...
      End Select
    EndIf
  EndIf
Next
```

Código 3. Chequeamos la pulsación del botón "OK"

```
DawImage botonB(38)\grafico,botonB(38)\x,botonB(38)\y
If ImagesOverlap(cursor_raton,MouseX(),MouseY(),botonB(38)\grafico,botonB(38)\x,botonB(38)\y)
  DrawImage boton(38)\grafico,botonB(38)\x,botonB(38)\y
  If MouseDown(1)
    c_sclic=PlaySound(sclic)
    ChannelVolume c_sclic,Volumen_SFX
    aceptar=1
  EndIf
EndIf
```

Texturas para bonos y sprites

Con esta entrega terminamos de realizar los gráficos para "Zone of Fighters".

Desarrollaremos las texturas para los bonos, lo cual nos servirá para aprender técnicas de solapamiento de capas para lograr más profundidad en texturas de modelos cúbicos. Además, aprenderemos a crear nuestros propios gradientes de color, los cuales son muy importantes para realizar texturas para sprites utilizados en sistemas de partículas.

TEXTURAS PARA LAS CARAS DE UN MODELO CÚBICO

Cuando tenemos un cubo y queremos adaptarle una misma textura en todas sus caras es importante tener en cuenta que los cuatro bordes de la textura coincidirán entre sí. También, es posible añadir efecto de relieve a cada cara aplicando una textura con múltiples capas superpuestas con efectos de sombreado para crear diferentes niveles.

En "Zone of Fighters"encontramos un ejemplo claro de este uso de texturas en los bonos que encontramos esparcidos por el terreno. Estos bonos son cubos girando en su eje Y mostrando la misma imagen en todas sus caras. Básicamente, la técnica utilizada para cada uno de los diferentes bonos es la misma. Por ello, solo vamos a explicar aquí las más significativas.

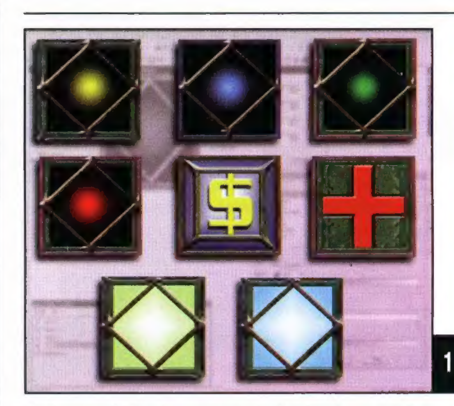
Para nuestro primer ejemplo vamos a realizar la textura de los bonos de munición de bajo calibre (Fig. 1).

CAPAS SUPERPUESTAS

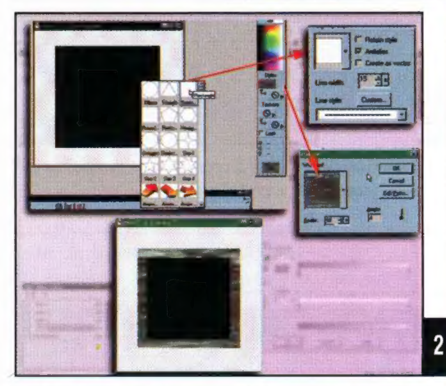
Para empezar creamos un documento nuevo de 128 x 128 de tamaño. La textura consta de diversos marcos superpuestos para dar sensación de profundidad. Para crear el primer marco lo primordial es crear una nueva capa, la cual llamamos "marco 1". La forma del marco la obtendremos a partir de la librería de *Preset Shapes* que utilizamos en el número anterior. Pulsamos en el icono de "Preset Shapes" y seleccionamos de la librería del panel de opciones la figura "Rectangle". Nos interesa solo el borde, así que seleccionamos la opción *Antialias* y colocamos un borde de 15 unidades en *Line Width*.

Antes de dibujar el cuadrado vamos a aplicarle una textura al borde para evitar un color plano. Nos dirigimos a la paleta *Colors* y seleccionamos el *Pattern Wrapped* para *Styles* de la librería de patrones. Ya podemos dibujar el marco sobre el fondo. Observamos cómo se crea un cuadrado con los bordes rellenos con el patrón elegido (Fig. 2).

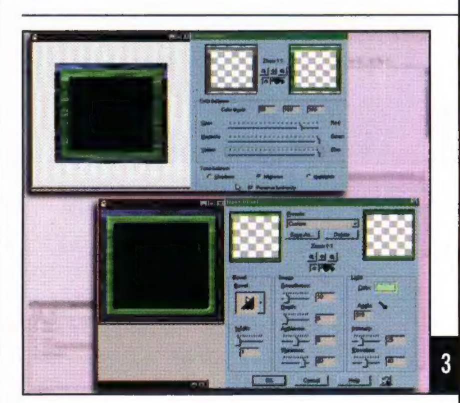
El siguiente paso es proporcionarle un color adecuado. Una forma muy fácil de hacer esto es utilizando el balance de color en *Colors / Adjust / Color Balance* con los valores -100, -100, 100 para obtener el tono azul. Hacemos las mismas operaciones para un segundo marco más pequeño, el cual dibujamos en una nueva capa que llamamos "marco 2". En esta ocasión el tono de color es verde. Al primer marco le aplicamos un efecto *Inner Bevel* para darle más apariencia de profundi-



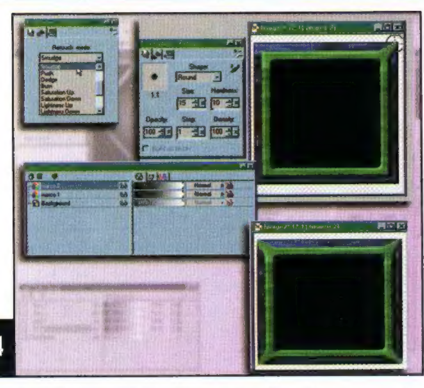
Imágenes de las texturas de todos los bonos del juego.



En la primera de las capas dibujaremos el marco situado más abajo.

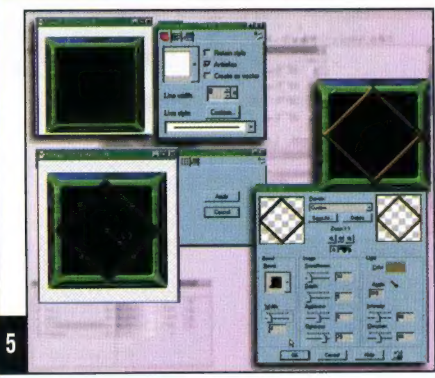


Aplicando el efecto Inner Bevel conseguimos más profundidad entre los marcos.



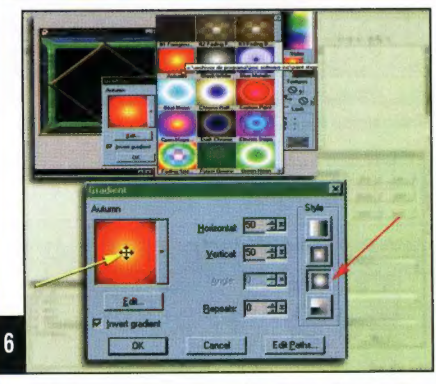
4

Con la herramienta de retoque Smudge conseguimos desplazar las esquinas del segundo marco para que parezca que está unido al marco inferior.



5

Ganamos mucha más profundidad si modificamos la orientación de una de los marcos.



6

Disponemos de cuatros estilos de gradientes diferentes: lineal, rectangular, esférico y radial.

dad con los siguientes parámetros:

- Figura primera de la librería en *Bevel* con un tamaño de 1.
- **En Image:** una suavidad de 10, profundidad de 5, ambiente a 0 y un brillo del 80%
- **En Light:** Color azul claro, ángulo a 315 grados, intensidad al 25 % y una elevación del 40%

Debemos recordar que a la capa del fondo no se le puede aplicar ningún efecto 3D de imagen. Para realizarlo, debemos duplicarla para crear una nueva capa con él y posteriormente, borrar por completo la capa del fondo.

Seleccionamos la capa del segundo marco y apliquémosle el mismo efecto con los mismos parámetros; es decir, aplicando *last used* en la casilla *presets*. Podemos observar cómo el marco más pequeño parece que está sobre el más grande (Fig. 3).

Aun así, sigue dando la sensación de que ambos marcos están separados. Así que una buena forma de unirlos es uniendo las esquinas del marcos pequeño con las del marco mayor. Por lo tanto, seleccionamos la capa "marco 2" y pasemos a alargar cada uno de los picos hacia las esquinas del dibujo. La mejor manera de realizar esta operación es utilizando la herramienta *Retouch* en el modo *Push* o *Smudge* (embadurnar), aunque es mejor la segunda. Elijámosla y apliquemos los siguientes parámetros al pincel de retoque: forma redonda, tamaño (*size*) 15, dureza (*hardness*) 10, opacidad y densidad a 100 y por último un paso (*step*) de 1. Una vez configurada la herramienta, nos situamos en una de las esquinas del marco pequeño verde y la desplazamos con cuidado hacia su misma esquina en la imagen (Fig. 4).

Después de realizar la operación con las cuatro esquinas, pasamos a aplicar un nuevo marco, pero esta vez rotado hasta adquirir la forma de rombo. Para dibujarlo, reducimos el grosor del borde a 8 unidades y elegimos el patrón *Leather* para *Styles*.

La rotación la aplicaremos con la herramienta de deformación. Al pulsar en el icono el marco se rodeará de pequeños cuadrados, los cuales sirven para modificar el tamaño de la imagen seleccionada. En el centro tenemos dos puntos más, unidos por una línea. El primero se utiliza para desplazar la imagen seleccionada y el segundo es el que utilizamos para rotar la figura. Colocamos el nuevo marco como se muestra en la figura 5.

Si una vez rotado observamos que la figura sobresale del resto de marcos, utilizamos los puntos de deformación de las esquinas para reducir el tamaño. Para aplicar la transformación pulsamos en el botón "Apply" de la ventana de opciones de esta herramienta *Toos Options – Deformation*. El siguiente paso será aplicar de nuevo un efecto *Inner Bevel* con los siguientes parámetros (Fig. 5):

- La quinta figura de la librería en *Bevel* con un tamaño de 4.
- **En Image:** una suavidad de 10, profundidad de 5, ambiente a 0 y un brillo del 80%.
- **En Light:** Color marrón claro, ángulo a 315 grados y una intensidad y elevación del 40%.

RECORDATORIO

Debemos recordar que a la capa del fondo no se le puede aplicar ningún efecto 3D de imagen. Para realizarlo, debemos duplicarlo para crear una nueva capa con él y posteriormente, borrar por completo la capa del fondo.

UTILIZACIÓN DE GRADIENTES DE COLOR

Para terminar nuestra textura vamos a crear la esfera de color amarillo del centro que representa el tipo de munición que proporciona este bono. Para lograr el efecto de esfera con un dibujo plano se suelen utilizar gradientes de color. Pero antes elegimos la herramienta de pincel con un tamaño de 70 y con la dureza (*Hardness*) a 0.

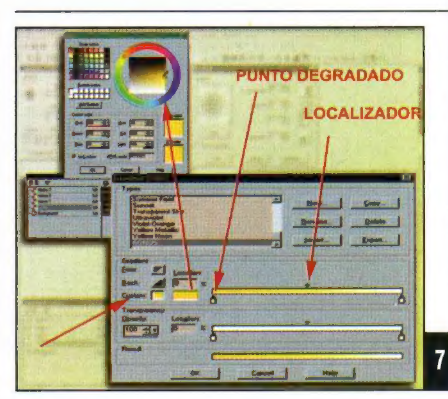
En Paint Shop Pro podemos crear los gradientes de color que queramos con cuatro estilos diferentes: lineal, rectangular, esférica y radial. Para seleccionarlos debemos elegir el estilo de gradientes en *Styles* pulsando sobre la flechita negra de *Styles* y eligiendo el segundo icono . A continuación, pulsando en el cuadrado de *Styles* aparecerá una ventana flotante con todas las opciones. A la derecha encontramos los cuatro estilos de gradientes. De arriba a abajo: lineal, rectangular, esférica y radial. En la pequeña ventana gráfica de la izquierda tenemos un previo del gradiente seleccionado. Podemos modificar el punto de tonalidad más clara modificando los parámetros "Horizontal" y "Vertical" o bien desplazando con el ratón la pequeña cruz que se muestra en la ventana gráfica de previo. Elegimos el estilo esférico y colocamos la cruz en el centro de la imagen (Fig. 6).

Para editar el aspecto del gradiente debemos pulsar en el botón "Edit". Aparecerá una nueva ventana llamada *Gradient Editor*. En la parte superior disponemos de una librería con algunos tipos de gradientes ya definidos pero que podemos añadir los nuestros propios. Los puntos de color y la distancia entre estos lo definimos en *Gradient*. La transparencia del gradiente la modificamos en *Transparency* y en *Result* se muestra el resultado final de mezclar *Gradient* y *Transparency*. En la figura 7 se muestra una descripción de estas secciones.

Para nuestra esfera vamos a utilizar un degradado de color desde el amarillo hasta el blanco. Así que vamos a crear uno nuevo para nuestro dibujo. Pulsamos en el botón "New" y le damos el nombre de "BONOS" al nuevo gradiente. En *Gradient* aparecerá un degradado entre un punto inicial y otro final. Estos puntos están representados por pequeños cuadrados rellenos de color. Pulsamos en el cuadrado de color situado a la derecha de *Custom* para activar el degradado por color. Para cambiar el primer tono a amarillo, hacemos clic cobre el primer cuadrado. El rectángulo de color situado a la derecha del botón "Custom" representa el color de este cuadrado. Hacemos clic sobre él para modificarlo. Seleccionamos un tono amarillo de la paleta de colores. Observamos cómo el primer cuadrado del gradiente se rellena con el color elegido. Hacemos lo mismo con el segundo punto del gradiente pero eligiendo un tono blanco. Hemos creado un gradiente de color desde el amarillo hasta el blanco. Podemos modificar el punto donde coinciden ambas tonalidades cambiando el valor de *Location* o desplazando el pequeño círculo sobre la línea de gradiente. Para nuestro dibujo situamos esta localización en el centro; es decir al 100% en *Location*. Pulsamos *Ok* para volver a la ventana *Gradient*. Antes de salir deseleccionamos la opción *Invert gradient*. Para terminar, seleccionamos la capa del fondo y hacemos clic con la herramienta de pincel previamente seleccionada en el

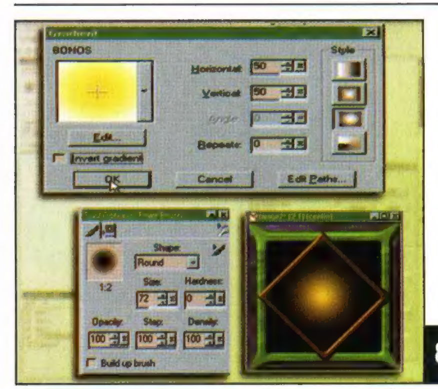
NOTA

En un gradiente circular el punto de color más claro indica dónde incide la luz.



7

A través del editor de gradientes (Gradient Editor) podemos definir cuántas divisiones de color queramos.



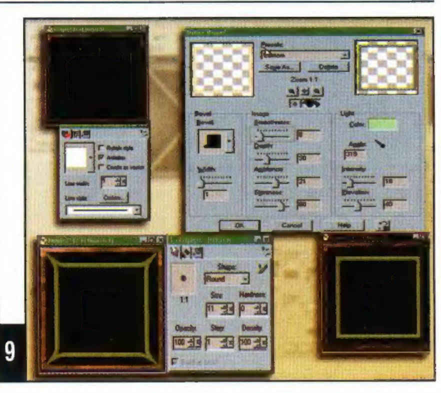
8

Cada herramienta de dibujo, como el pincel o el bote de relleno, puede tener su propio gradiente. Así que es importante seleccionarla previamente antes de elegir un gradiente.

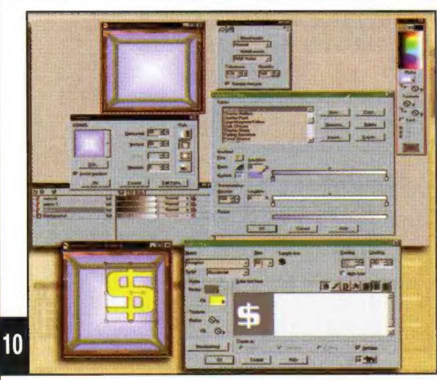
DEFINICIÓN

GRADIENTE

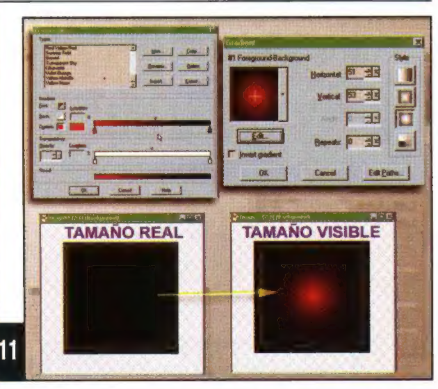
Un gradiente consiste en colocar un tono de color en un punto de la imagen y otro tono diferente en otro punto y extender uno hacia el otro. Con esto se consigue rellenar el espacio entre los dos puntos con variaciones de color.



9 *Procedimientos para dibujar el marco interior y unirlo al marco exterior.*



10 *Utilizar un gradiente para rellenar el hueco entre los dos marcos simula el aspecto interior del modelo cúbico.*



11 *Para evitar la forma cuadrada de la textura cuando visualicemos el sprite en el juego, debemos dibujar el gradiente más pequeño que el tamaño total de la imagen.*

centro de la imagen (Fig. 8).

Vamos a realizar otra de las texturas para practicar un poco más los procedimientos anteriores. Para ello, elegimos el bono de puntuación, que como podemos ver en la figura 1, consta de dos marcos, un gradiente rectangular y una letra sombreada.

Empezamos con un nuevo documento de 128 x 128. Creamos una nueva capa y dibujamos en ella el primero de los marcos de la misma manera que lo hicimos para la textura anterior y le aplicamos el efecto *Inner Bevel* que guardamos en *presets*. El segundo marco lo colocamos en el interior del primero. Le asignamos un borde de 5 píxeles y lo dibujamos en una nueva capa. De nuevo, le aplicamos un efecto *Inner Bevel* pero con los siguientes parámetros:

- La quinta figura de la librería en *Bevel* con un tamaño de 1.
- **En Image:** una suavidad de 8, profundidad de 30, ambiente a 21 y un brillo del 80%.
- **En Light:** Color marrón claro, ángulo a 315 grados, una intensidad del 18% y una elevación del 40%.

Seguidamente, con la herramienta de retoque (*Retouch*) elegimos el modo *Smudge* y asignamos un tamaño de 10 al pincel de retoque. Nuestra intención es desplazar las esquinas del marco interior hasta las esquinas del marco exterior para unirlas (Fig. 9).

Para aumentar el efecto de profundidad, pasemos a rellenar el fondo negro con un gradiente rectangular. Seleccionamos la herramienta de relleno y cambiemos en *Styles* el color de tinta por un gradiente. En la ventana *Gradient* elegimos el estilo rectangular. Entramos en el editor de gradientes y creamos uno nuevo. Para el primer punto del gradiente elegimos un tono azul y para el segundo el color blanco. Aceptamos y en la ventana *Gradient* seleccionamos la opción *Invert Gradient*. Antes de salir, colocamos el punto de tono de color más claro en el centro de la imagen de previo (si no lo está). Rellenamos la capa del fondo con el gradiente, haciendo clic en el centro de la imagen.

Ya solo nos queda colocar el símbolo " $ " en el centro de la textura. Para ello, seleccionamos la herramienta de texto. Elegimos el tipo de letra que creamos para el juego ("Dungeon"); en su defecto podría valer "Arial" o "Times". Seleccionamos un tamaño de letra de 80 y elegimos un color amarillo en *Fill*. Situamos el símbolo en el centro de la imagen. Observamos cómo se ha creado una nueva capa vectorial. Para aplicar una sombra hay que convertirla a Bitmap rasterizándola (*Convert to Raster Layer*) (Fig.10).

Para terminar, le aplicamos el efecto 3D *Drop Shadow* con un *Offset* de 2 (vertical) y -1 (Horizontal) y unos atributos con opacidad 73 y desenfoque (*blur*) del 8,5 %.

SPRITES PARA SISTEMAS DE PARTÍCULAS

Los gradientes de color son muy utilizados para crear las texturas para sprites utilizados en sistemas de partículas o disparos. Para realizarlas, lo más importante a tener en cuenta es que el gradiente nunca llegue a cubrir toda la imagen; es decir, que no llegue al límite de su tamaño. Si no lo hiciéramos así, durante el juego veríamos, además del gradiente, el cuadrado de la textura. En la figura 11 se muestra cómo se diseña una textura para una partícula esférica.



En el próximo número...

... aprenderemos otras técnicas de diseño.

Secuenciadores trackers (II)

Para ilustrar el funcionamiento de ModPlug vamos a realizar una canción simple con dos o tres patrones.

La idea que perseguimos es que el lector pueda empezar a trabajar con este secuenciador y obtener resultados útiles.

PRIMEROS PASOS

Lo primero que tenemos que hacer es crear una canción nueva y elegir el formato que vamos a utilizar. Solo usaremos instrumentos basados en muestras de audio, así que la mejor opción para empezar es utilizar el formato de Impulse Tracker (IT). Por lo tanto, ejecutamos el programa y nos vamos al menú *File* para elegir la opción *New* ("CTRL"+ "N") y la opción "IT".

Aparecerá una nueva ventana en la zona de edición sin título. Agrandémosla al máximo para adaptarla a toda la zona. Nos encontramos en la pestaña *General*. Observamos cómo en *Song Type* se muestra el formato IT con 16 canales. Como solo vamos a utilizar cuatro, cambiemos el número de ellos pulsando en *Change* y eligiendo *4 Channels*. El siguiente paso será dar nombre a la nueva canción en la casilla *Song Title*. Por el momento mantenemos el tempo por defecto de 125 negras por minuto. Pasemos a la zona inferior para renombrar cada uno de los 4 canales como "C1", "C2", "C3" y "C4" (Fig. 1).

El resto de parámetros los dejamos como están. Pasemos a la sección de patrones pulsando en la pestaña *Patterns*. Podemos ver cómo se muestran los cuatro canales y un patrón con 64 líneas del primero.

Demos un nombre al patrón, por ejemplo "PATRON 1" en la casilla *Pattern name*. Antes de poder tocar cualquier nota es necesario cargar un sonido. En la casilla *Instrument* es donde se selecciona cada sonido. Por el momento no hay ninguno.

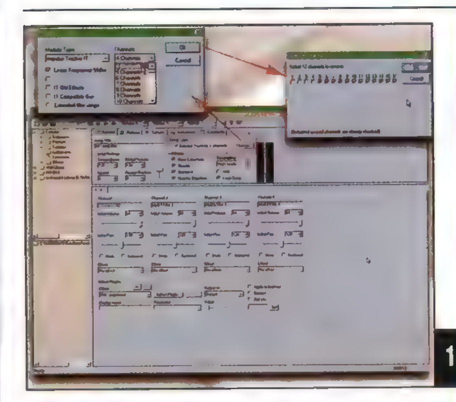
CARGANDO SONIDOS

Para cargar los sonidos pasemos a la casilla *Samples*. Por defecto disponemos de un sonido vacío llamado "untitled". Pulsamos en el icono para sustituir el sonido por uno nuevo. En el directorio "Extras" del CD se ha proporcionado una selección de ellos. Elegimos "C_Kick", el cual corresponde a un bombo de batería. Observamos cómo "untitled" es sustituido por el nombre del nuevo sonido y en la parte inferior de la pantalla aparece su representación gráfica. Para reproducirlo podemos pulsar en el icono o bien a través del teclado (Fig. 2).

El teclado del ordenador nos sirve para reproducir los sonidos. Cada tecla representa un tono diferente desde "Q a +", "A a ´ " y "Z a –" (Fig. 3).

Podemos reproducir solo una parte del sonido seleccionándola con el ratón en la gráfica. Además, es posible tocarlo varias veces a la vez pulsando varias teclas simultáneamente como si fuera un acorde.

Para añadir un nuevo sonido debemos pulsar en el icono . Al hacerlo la casilla *Sample* mostrará un "2" indicando que estamos en una nueva platilla a la espera de cargar una nueva muestra. Elijamos entonces "C_HC" del disco, el cual corresponde a una caja. Hacemos la misma operación con el sonido "Maracas" y con "Vangelis". Bien, ya estamos preparados



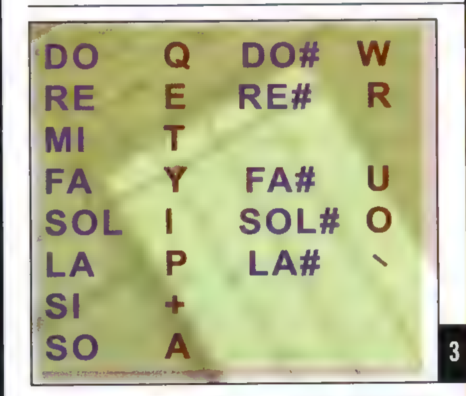
1

Podemos ajustar el número de canales que vamos a usar en Song Type.



2

Para sumar sonidos debemos crear una plantilla nueva y luego cargar el fichero de audio.

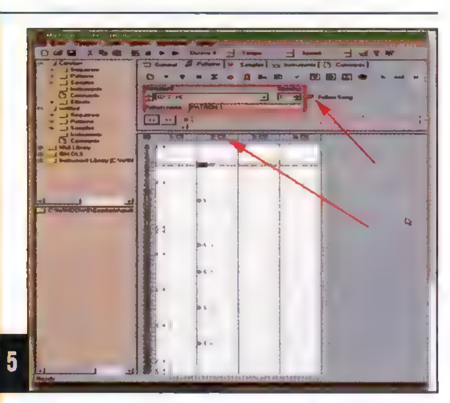


3

Esquema de la equivalencia en el teclado del ordenador de la escala cromática.

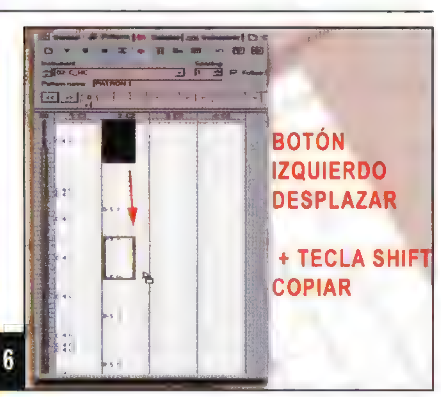
4

Podemos grabar nota a nota o bien mientras se reproduce el patrón.



5

Podemos aplicar varios sonidos diferentes en una misma columna.



6

Mediante el uso del ratón podemos copiar o desplazar grupos completos de notas de un lugar a otro del patrón.



TRUCO

ModPlug dispone de varios niveles de "UNDO" con "CTRL + Z".

para escribir las primeras notas.

EDITANDO EL PATRÓN

Pasemos a la sección de patrones. Para navegar por el patrón es bien sencillo. Solo tenemos que utilizar los cursores o el puntero del ratón. Con "CTRL + INICIO" nos situamos en la primera línea y con "CTRL + FIN" en la última. Pulsando en la tecla "INICIO" nos situamos al principio de la primera columna y con "FIN" al final de la última. Con la tecla de tabulación pasamos de una columna a otra. Si disponemos de un ratón con rueda, desplazaremos la ventana de patrones verticalmente con ella de manera rápida. Elegimos de la casilla *Instrument* el primer instrumento que cargamos, "C_Kick". Podemos realizar esta operación de varias formas, bien desde la misma casilla o con "CTRL + Cursor arriba" y "CTRL + Cursor abajo".

En este momento, si pulsamos el teclado oímos el sonido elegido. Para poder colocar nuevas notas en el patrón a medida que tocamos es necesario dejar pulsado el icono de grabación. Nos situamos al principio del todo del patrón en la parte para las notas de la primera columna. Pulsamos en el teclado la "Q", correspondiente al DO de la cuarta octava (C4) de un teclado musical. Vemos que aparece "C-4" y al lado "01"; es decir, correspondiente al tono y al número del instrumento elegido, y que el cursor se sitúa en la siguiente línea automáticamente, a la espera de una nueva grabación (Fig. 4). Podemos grabar mientras se reproduce el patrón pulsando también en el botón de reproducción. Este método es menos real ya que la velocidad de grabación no corresponde con la de reproducción. Por lo tanto, es preferible situarnos en la línea manualmente y tocar las notas una a una. Situemos el cursor en la línea 0 y pulsamos la tecla "Q". Seguidamente hacemos lo mismo para las líneas siguientes: "8, 16, 18, 19, 24, 32, 40, 48, 50, 51 y 56".

Es posible copiar y pegar una nota o un grupo de ellas con solo seleccionarla con el ratón y utilizando el método de copiar y pegar ("CTRL+C" y "CTRL+V").

Si no hemos colocado una nota correctamente y queremos borrarla, solo tenemos que hacer clic sobre ella y pulsar la tecla "SUPR". Si queremos borrar toda la línea, podemos seleccionarla con el ratón o a través de la opción *Delete Row* (tecla de borrar) del menú flotante que aparece al hacer clic con el botón derecho del ratón sobre la línea. Hay que tener en cuenta que al suprimir una línea todas las que le siguen se situarán en una línea anterior.

Aunque no tengamos ningún volumen definido para cada nota, éste por defecto se coloca en el valor mayor posible. De todas formas, es posible aumentarlo más si cabe con el botón de amplificación y eligiendo un porcentaje mayor del 100%.

Bien, ya tenemos el primer canal con música. El siguiente paso será grabar la caja. Nos situamos en el canal segundo (2:C2) y seleccionamos el sonido número 2 (C_HC) correspondiente a la caja. Hagamos la misma operación anterior y coloquemos la nota D5 correspondiente a la tecla "D" en las siguientes líneas: "4, 12, 22, 28, 36, 44, 54 y 60" (Fig. 5).

Podemos desplazar grupos de notas de una posición del patrón a otra (a cualquier canal si queremos) seleccionándolas con el ratón y desplazándolas sin dejar de hacer clic. Para copiar el grupo seleccionado tenemos que mantener pulsada también la tecla "SHIFT" (Fig. 6).



En el próximo número...

... terminaremos nuestro pequeño ejemplo añadiendo nuevas notas y aplicando efectos.

Funciones de red y protocolos (II)

En el número anterior nos quedamos en el funcionamiento de los comandos para la comunicación con el protocolo UDP.

En esta entrega terminaremos de ver las funciones UDP, así como el manejo de las DirectPlay.

FUNCIONES PARA GESTIONAR EL PROTOCOLO UDP (Y II)

Como sabemos, en toda comunicación UDP es necesario conocer el puerto de quien recibirá el mensaje, pero también necesitamos saber su dirección IP. El valor de la IP se obtiene directamente al recibir un mensaje con "RecvUDPMsg", ya que esta función permite leer los datos del "stream" y además, retorna un número entero correspondiente a la IP de quien lo envía:

```
Mensaje = RecvUDPMsg ([illegible])
IP_Cliente = Mensa[illegible]
```

Sin embargo, existe otra manera de averiguar este dato sin la necesidad de abrir el "Stream" para leerlo con "RecvUDPMsg" y es con la instrucción "UDPStreamIP":

```
IP Stream = UDPStreamIP
([illegible])
```

A menudo, necesitaremos saber el puerto y el IP pero del último mensaje recibido. Para ello disponemos de las funciones: "UDPMsgPort", para averiguar el puerto, y "UDPMsgIP" para la IP.

Al igual que ocurre en el protocolo TCP, es importante poder controlar la actividad de las funciones que reciben la información de la red. En el caso del protocolo UDP podemos determinar el tiempo que la función "RecvUDPMsg" esperará para recibir un nuevo mensaje con la instrucción "UDPTimeouts":

```
UDPTimerouts [illegible]
[illegible]
```

Por defecto, el valor de espera vale 0, así, si "RecUDPMsg" no recibe nada, la función terminará inmediatamente.

OBTENCIÓN DE LA DIRECCIÓN IP

Generalmente, el usuario no conoce la dirección IP de su equipo o de cualquier otro; es posible obtenerla desde Blitz3D con la función:

```
"HostIP": HostIP ([illegible])
```

El parámetro "Número del Host" se refiere a un "host" (anfitrión) determinado de todos los disponibles. Para averiguar este dato disponemos de la función "CountHostIPs", la cual almacena en una variable el número de "Hosts" conocidos:

```
[illegible] = " "
[illegible] = CountHostIPs
([illegible])
```

Si a "Nombre_Host$" no se le pasa ningún nombre contendrá el del "Host" local.

Por lo tanto, para obtener la IP de un equipo local:

```
[illegible] = HostIP(CountHostIps(""))
```

Sin embargo, la variable "Num IP" contendrá como valor de la IP un número ente-

1

OBTENER IP

IP = RecvUDPMsg
IP = UDPStreamIP

OBTENER PUERTO

Puerto = UDPMsgPort

Procesos para obtener la IP y el puerto del que envía un mensaje utilizando el protocolo UDP.

2

OBTENER IP LOCAL

Num_Host = CountHostIP ("")
Num_IP = HostIP (Num_Host)
IP$ = Dotted$ (Num_IP)

Procesos básicos para obtener la IP con notación estándar 1.2.3.4.

NOTA

SIGNIFICADO DE LOS NÚMEROS DE UNA IP

- Primer número: **País o zona geográfica**
- Segundo número: **Organización o provincia**
- Tercer número: **Grupo de Ordenadores**
- Cuarto número: **Terminal**

ro, que es la manera interna que Blitz3D tiene de manejarla. Recordemos que la notación que nosotros manejamos de una dirección IP se compone de un grupo de cuatro valores numéricos desde 0 hasta 255 separados por un punto. Así que debemos convertir el valor entero contenido en "Num_Host" a esta notación mediante la función:

```
"DottedIP$": DottedIP$ (    )
```

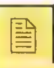
Así que podemos añadir:

```
= DottedIP$ (    )
```

(Ver "Ejemplo19_1.bb")

FUNCIONES PARA EL MANEJO DE LAS DIRECTPLAY

Aprovechando la oportunidad que Blitz3D nos brinda para manejar las DirectPlay por medio de una serie de funciones específicas, podemos crear y gestionar partidas multijugador con suma facilidad. El uso de las DirectPlay no es recomendable para comunicar a más de dos jugadores para juegos en tiempo real debido a su lentitud. Sin embargo, es ideal para crear juegos en donde no es necesaria una gestión inmediata de la información, por ejemplo, en juegos por turnos como los de tablero, cierto tipo de estrategia, etc. Otra gran desventaja que podemos encontrar en el uso de las DirectPlay es que es necesario que haya conexión entre al menos dos equipos y que haya una partida y al menos un jugador creados de antemano para que funcione.



NOTA

El uso del protocolo UDP para un juego multijugador puede funcionar sin necesidad de tener al menos dos equipos conectados o una partida creada como ocurre con las DirectPlay.

¿CÓMO CREAR UNA NUEVA PARTIDA MULTIJUGADOR O UNIRSE A ELLA?

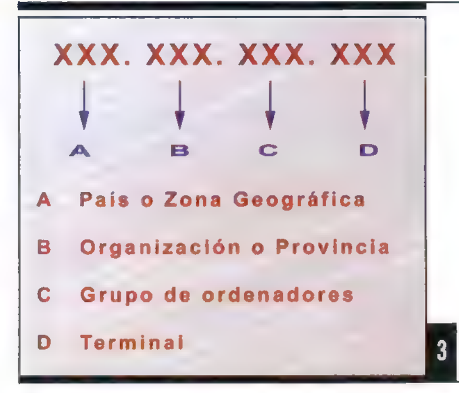
Antes de comenzar cualquier partida multijugador es necesario crearla en un equipo que actuará de anfitrión (Host) y al cual se unirán los demás jugadores. Podemos realizar esto de dos formas diferentes: Utilizando la ventana de diálogo estándar de Windows o bien directamente.

Para el primer método debemos usar la función:

```
"StartNetGame()":
              = StartNetGame()
```

Esta función también nos sirve para unirnos a una partida ya creada. Al terminar de ejecutarse se devolverá en "Nuevo_Juego" un valor indicando el resultado de la operación realizada por "StartNetGame()". Si el valor es 0 indica que no se ha podido crear un nuevo juego. Si es 1, que nos hemos unido a un juego existente y si devuelve un 2, es que se ha creado un nuevo juego (Ver Código 1).

Para el segundo de los métodos, es decir, prescindiendo de la ventana de diálogo de Windows, debemos utilizar "HostNetGame" para crear un nuevo juego (devuelve un 2 si se crea el juego):



Significado de cada uno de los grupos de números que componen una dirección IP.

```
HostNetGame (Nombre de la partida $)
```

y "JoinNetGame" para unirnos a un juego existente (devuelve un 1 si se une al juego).

```
JoinNetGame (Nombre de partida,
IP del host o servidor)
```

El segundo parámetro se refiere a la dirección IP en notación estándar 1.2.3.4.

Por lo tanto, un ejemplo para crear una partida sería:

```
              = HostNetGame
("         ")
If              =
    Print "
                    "
Else
    Print "
          "
Endif
```

Y para unirnos a ella sería:

```
        = "         "
              = JoinNetGame
("PARTIDA 1", IP_Host )
If              =
    Print "
                    "
  Else
    Print "Unión al
```

Código 1. Vemos el resultado de la operación que realiza "StartNet Game"

```
Nuevo_Juego = StartNetGame()
If Nuevo_Juego = 0 Print "No se ha podido crear un nuevo juego o unir a uno existente"
If Nuevo_Juego = 1 Print "Se ha unido al juego correctamente"
If Nuevo_Juego = 2 Print "Nuevo juego creado correctamente"
```

```
[illegible]"
Endif
```

Una vez creada una partida con "StartNetGame ()" es posible pararla utilizando "StopNetGame ()". Esto no implica que la partida se elimina completamente ya que es posible transferir la sesión del host a otro equipo conectado al juego, con lo que la partida continuaría.

GESTIONANDO UNA PARTIDA MULTIJUGADOR. LOS JUGADORES

Toda nueva partida debe tener jugadores. Además, es necesario tener al menos uno creado para poder enviar o recibir mensajes Así que para crear un nuevo jugador disponemos de la función que podemos ver en el Código 2.

La variable "Nuevo_Jugador" contendrá un número entero con la identificación interna del jugador, el cual es usado en las operaciones de envío y captación de mensajes. Como veremos más adelante, también se produce un mensaje especial que es enviado a todas los demás jugadores (equipos remotos) avisando de la nueva creación. Lo mismo ocurre si borramos un jugador, lo cual podemos realizar aprovechando el número de identificación del mismo por medio de la función "DeleteNetPlayer":

```
DeleteNetPlayer ([illegible]
[illegible])
```

Un ejemplo sería:

```
Nuevo Juego = StartNetGame()
Nuevo Jugador = CreateNetPlayer
("Dario")
DeleteNetPlayer Nuevo_Jugador
```

Una vez que hemos creado los jugadores es posible que cualquiera de ellos pueda saber el nombre de los demás utilizando "NetPlayerName$":

```
Nuevo Juego = StartNetGame()
Nuevo Jugador = CreateNetPlayer
("Dario")
Print "El nombre d[illegible]
"+ NetPlayerName$ ([illegible]
```

Podemos obtener más específicamente, además del nombre, si el que ha enviado el mensaje es el jugador del "host" con la función "NetPlayerLocal", si es así, devolverá el valor 1 y si no, un 0.

GESTIONANDO UNA PARTIDA MULTIJUGADOR. LA INFORMACIÓN

Después de crear un juego y los jugadores utilizando las DirectPlay debemos gestionar la información que transita entre ellos a través de la red. Para recibir los mensajes que llegan por la red debemos utilizar "RecvNetMsg()" continuamente, ya que lo único que hace es chequear si ha entrado un mensaje. En caso afirmativo devuelve un valor "True":

```
While Not KeyHit(1)
  If RecvNetMsg() = True Print
  "[illegible]"
Wend
```


Mensajes devueltos al iniciar un juego multijugador con la instrucción "StartNetGame" correspondiente a la ventana estándar de Windows para DirectPlay.

Cada mensaje enviado o recibido lleva incluido, además del mensaje propiamente, cierta información que indica su procedencia y su tipo. Blitz3D proporciona diferentes funciones para aislar y obtener cada uno de estos datos.

Supongamos que un jugador de la partida ha recibido un nuevo mensaje, puede saber de qué tipo es con la función "NetMsgType()". En la

Proceso básico para crear un nuevo jugador utilizando las DirectPlay desde Blitz3D.

Código 2. Creamos un nuevo jugador

```
"CreateNetPlayer":
Nuevo_Juego = StartNetGame()
Nombre_Jugador$ = "Darío"
Nuevo_Jugador = CreateNetPlayer (Nombre_Jugador$)
If Nuevo_Jugador = 1
    Print "Se ha creado el jugador "+Nuevo_Jugador+" con el nombre "+Nombre_Jugador$
  Else
    Print "No se ha podido crear el jugador"
Endif
```

1 - 99	Mensaje de usuario
100	Nuevo jugador
101	Jugador borrado
102	Host inicial deja el juego
200	Error grave

6

Tipos de mensajes devueltos por "NetMsgType()".

figura 6 se muestra una tabla con los diferentes tipos de mensajes posibles.

Por ejemplo, si el valor devuelto por esta función es un 100 significa que un nuevo jugador se ha unido a la partida.

Para saber quién ha enviado el nuevo mensaje podemos usar "NetMsgFrom", el cual devolverá su número de identificación (el que le fue asignado cuando se creó con "CreateNetPlayer"). Esta información es importante si queremos que el host actúe de alguna manera con cierto jugador remoto.

Igualmente, podemos obtener información de a quién va dirigido un mensaje utilizando "NetMsgTo()". Y por último, para obtener el mensaje en sí usamos "NetMsgData$()". Esta función devolverá un string conteniendo el mensaje:

```
StarNetGame()
[illegible]evo_Jugador = CreateNetPlayer
("Darío")
While Not KeyHit(1)
  If RecvNetMsg()
    [illegible]po_Mensaje = NetMsgType ()
    Print "EL MENSAJE ES:
    "+NetMsgData$()
    Print "Y LO EN[illegible]
    "+NetMsgFrom()
    Print "Y FUE [illegible]:
    "+NetMsgTo()
  Endif
Wend
```

Para terminar, la función que nos permite enviar un mensaje a cualquier jugador, "SenNetMsg":

```
SenNetMsg [illegible]
[illegible]
[illegible]
```

El parámetro "Tipo de mensaje" debe valer entre 1 y 99 e indica que es un mensaje de usuario (diferente a los explicados anteriormente).

- **"Mensaje$"** es un string conteniendo el mensaje, propiamente dicho, que queremos enviar.
- **"ID del Jugador"** es el número identificativo del jugador que envía el mensaje.
- **"Jugador de Destino"** es la identificación del jugador que recibirá el mensaje. Si colocamos un valor 0 se enviará a todos los jugadores.
- **"Prioridad"**. Este parámetro permite evitar que no se pierda ningún paquete de información. Si se coloca algún valor mayor de 0 indicará el grado de certeza en el envío de los mensajes. Evidentemente, esto conlleva a una mayor lentitud. Si no se coloca ningún valor para este parámetro se asume el riesgo de pérdida de información pero se gana velocidad.

Un hecho importante, el cual ya hemos comentado, es la baja velocidad a la que trabaja el tránsito de información mediante las DirectPlay. Un buen truco para evitar al máximo la pérdida de velocidad es colocar en un mismo string toda la información que se quiere enviar de una vez; es decir, si queremos mandar las coordenadas (300, 200) y el valor estado (2) de un jugador al resto de jugadores, incluimos estos valores en una misma línea y la enviamos:

```
[illegible] ="[illegible]"
```

Luego, al ser recibido, se extraería cada uno de los valores de la línea obtenida mediante funciones de cadenas. En el ejemplo 7 se muestra el esquema de esta técnica.

X = 350 : Y = 200 : ESTADO = 1

MENSAJE " 350, 200, 1 "

ENVIAR MENSAJE

" 350, 200, 1 "

X Y ESTADO

7

Esquema del envío de numerosos datos en una sola línea de texto para ganar velocidad en la transferencia de información utilizando funciones para DirectPlay.

```
PosBank=CreateBank (4)
; ..................................
Function FloatS$(valor#)
 PokeFloat PosBank,0,valor
 For n=0 To 3
  S$=S$+Chr(PeekByte(PosBank,n))
 Next
 Return S$
End Function
```

8

Función para convertir un número float en string para la comunicación por TCP. Es utilizada para el envío del mensaje.

```
PosBank=CreateBank (4)
; ..................................
Function StringF#(St$)
 For n=0 To 3
  PokeByte PosBank,n,Asc(Mid(St$,n+1,1))
 Next
 Num#=PeekFloat (PosBank,0)
 Return Num
End Function
```

9

Función para convertir el contenido de un string en un número float para la comunicación TCP. Es la función contraria de la figura 8 y es utilizada en la recepción del mensaje.



En el próximo número...

... finalizaremos hablando sobre la ampliación del lenguaje.

Realización de un vídeo para nuestro juego (y IV)

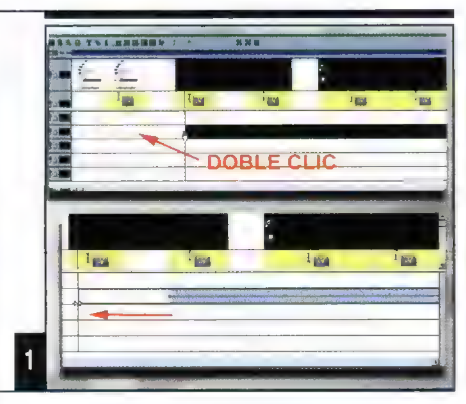
En este número terminaremos la realización del vídeo para nuestro juego con Ulead MediaStudio, montando las pistas de audio. Posteriormente, añadiremos algunos efectos de titulación y croma.

LAS PISTAS DE AUDIO

Como sabemos, la ventana de montaje o "timeline" se encuentra dividida en dos partes: la parte superior es para el vídeo y la inferior para el audio. Como sucede en la sección de vídeo, la de audio se encuentra divida por pistas. Cada una de ellas puede contener un fichero de audio en mono o en estéreo.

CARGAR Y SITUAR UN FICHERO DE AUDIO

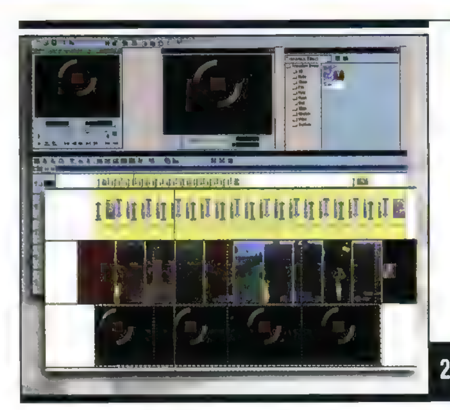
Para insertar un archivo de sonido solo tenemos que hacer doble clic sobre cualquiera de las pistas de audio. Podemos cargar tanto .WAV como .MP3 o .AIF. Al obtener el fichero aparecerá junto al puntero del ratón un rectángulo negro que representa el nuevo clip de audio. Este rectángulo tendrá la longitud equivalente a la duración del audio cargado. Haciendo clic sobre cualquier lugar de una pista lo situamos (Podremos encontrar el audio de la presentación en el directorio "Extras" del CD).
Desplazando los extremos podemos reducir o aumentar el tamaño del audio pero solo cuando se reproduce; es decir, esto no significa que cortemos el fichero sino que seleccionamos lo que queremos que se oiga. Para desplazar el audio solo tenemos hacer clic sobre la parte azul del rectángulo y mantener pulsado el botón izquierdo del ratón mientras lo desplazamos (Fig. 1).



Podemos cargar cuantos ficheros de audio queramos y colocarlos en cualquier canal.

CAMBIAR EL VOLUMEN DEL AUDIO

Un método muy importante y que resulta imprescindible utilizar para lograr una buena implementación del audio a una película es realizar ajustes en su volumen. Esto nos permitirá mezclar un sonido con otro acorde al desarrollo del vídeo. Para nuestra presentación utilizamos dos músicas diferentes y un efecto de audio de una explosión. La primera de las músicas entrará en acción justo cuando el logotipo del desarrollador funde a negro. La entrada de este audio se realiza desde un volumen mínimo. Si observamos en el clip de audio vemos que está dividido horizontalmente por una delgada línea cuyos extremos terminan en un pequeño cuadrado. Esta línea representa el nivel de volumen del clip y desplazando los cuadrados con el ratón podemos modificarlo. Así que, para lograr un "Fade In" de la música, debemos crear un punto de ruptura en esta línea para poder modificar el volumen. Para ello, hacemos clic sobre la línea en el lugar donde queremos crear el punto. Vemos que aparece un pequeño cuadrado igual que el de los extremos. Haciendo clic sobre él y desplazando el ratón podemos modificar su posición. Este nuevo punto marcará el punto más alto de volumen. Para crear el más



A partir del canal B, los demás canales se mezclan con opciones de overlay.

TRUCO

Si hacemos clic sobre el audio cargado se mostrará gráficamente en la ventana *source*.

NOTA

Para salvar el vídeo incluyendo el audio: después de elegir *File/Create/Video File* pulsamos en el botón "Options" y en la pestaña *General* seleccionamos *Audio and Video* en la


De forma sencilla MediaStudio nos permite añadir titulación en nuestra película con mezcla automática.

bajo solo tenemos que desplazar hacia debajo el cuadrado del extremo izquierdo. La línea asciende desde el primer cuadrado al segundo que hemos creado. Podemos añadir cuantos puntos de ruptura queramos y por lo tanto, cuantos cambios de volumen necesitemos.

Para cada clip de audio podemos modificar su velocidad de reproducción, duración e invertirla, eligiéndolo en el menú que aparece haciendo clic con el botón derecho del ratón sobre el clip ("Speed", "Duration" y "Reverse").

AÑADIENDO TITULACIÓN Y CROMA KEY

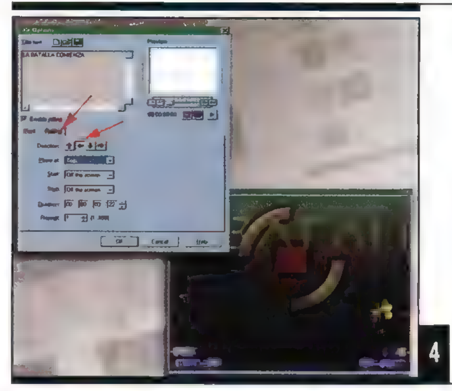
Aunque hemos trabajado con los canales A y B de vídeo podemos utilizar el resto para añadir algún tipo de titulación o efectos de *croma key*.

Vamos a añadir en la parte donde se muestran las capturas el logotipo del juego, continuamente mezclado, el resto de la imagen (efectos incluidos). Para ello, debemos utilizar un efecto de *croma key* para hacer que el color negro del logotipo se haga transparente, algo parecido a lo que ocurre con el canal alfa de una imagen. De esta forma, conseguimos que se muestre la imagen del logotipo sobre la película. En primer lugar, importamos el archivo de imagen y lo colocamos en el cuarto canal de vídeo, justamente debajo del canal B. Seguidamente lo extendemos hasta que ocupe toda la parte del montaje donde se muestran las capturas. Si hacemos un previo podemos observar que solo podemos ver la imagen del logo (Fig. 2)

Para activar el *croma key* elegimos la opción *Clip/Overlay Options* ("CTRL" + "R"). Aparece una ventana de diálogo en donde seleccionamos en la casilla *Type* la opción *Chroma Key*. A continuación, debemos elegir el color que servirá de croma. Hacemos clic en el cuadrado negro situado en la parte inferior izquierda. Seguidamente, en la ventana *Overlay clip* hacemos clic con el cursor (convertido en cuentagotas) sobre el color negro de la imagen. Podemos observar los resultados haciendo clic en el icono de "play" de la ventana *Overlay preview*. Para evitar los dientes de sierra de los bordes del logotipo podemos aplicar un pequeño desenfoque eligiendo la opción *Small* en la casilla *Soft edge*.

Podemos mezclar con todo el montaje anterior (*croma key* incluido) algún tipo de titulación. Queremos añadir la frase "LA BATALLA COMIENZA" atravesando horizontalmente la pantalla. Para ello, seleccionamos un clip de texto en *Insert/Tittle clip* y escribimos la frase con mayúsculas en la ventana de texto. En la pestaña *Font* elegimos nuestro tipo de letra del juego con un tamaño de 60, en *Fill color* elegimos un tono amarillo. Aceptamos en "OK" y situamos la imagen en un nuevo canal debajo del logotipo (Fig 3).

De nuevo entramos en las opciones con "CTRL" + "R". Para que se muestre transparente en la pantalla colocamos en la casilla *Text transparency* un 75. Pulsamos en la pestaña *Rolling* y activamos la flecha de dirección hacia la izquierda y en las casillas *Start* y *Stop* seleccionamos *Off the screen* para situar la frase fuera de la pantalla al principio y al final. Podemos ver el resultado activando los dos íconos de monitor de la ventana *preview* y haciendo play (Fig. 4).


La pestaña Rolling nos permite añadir a nuestra titulación diferentes tipos de movimiento.

DEFINICIÓN

CROMA KEY

Un *croma key* consiste en sustituir un color determinado de la imagen por otra imagen. Un claro ejemplo lo encontramos en los mapas del tiempo donde el presentador se encuentra delante de un fondo azul. Luego, este fondo azul es sustituido por la imagen del mapa formando la composición final.

NOTA

El orden de prioridad en la visualización de canales de vídeo es de abajo a arriba; es decir, el canal primero estará por detrás del segundo y así sucesiva-


En el próximo número...

... prepararemos la instalación del juego con Inno Setup.

Simuladores de velocidad

Este tipo de juegos siempre ha estado vinculado a las recreativas, ya que precisan de ciertos dispositivos como volantes, pedales o manillar para obtener los mejores resultados de jugabilidad, los cuales prácticamente no existían para los primeros equipos de juegos domésticos.

Básicamente, la finalidad que se persigue es simular una carrera de cualquier tipo bien con automóviles, motos, naves espaciales, etc. Podemos encontrar multitud de títulos con este concepto. Sin embargo, solo veremos las más importantes que son: automóviles, motociclismo y de tipo futurista.

HISTORIA Y EVOLUCIÓN TÉCNICA. AUTOMÓVILES

La simulación de autos de carreras es la más usada por los desarrolladores a lo largo de la historia. Sin lugar a dudas, la competición más veces recreada en un ordenador es la Fórmula 1.

FORMULA 1/ INDY

Ya existían simuladores de carreras de automóviles en el año 1977 para el antiquísimo sistema *RCA Studio II*. No fue hasta la década de los 80 cuando este género llegó al PC. Por aquel entonces triunfaba un juego de F-1 llamado *Pole Position* (Arcade) para recreativas. Este título se versionó en casi todas las plataformas existentes. Para PC – DOS apareció de manos de Synergistic Software en 1983. Básicamente, se trataba de correr contra el ordenador y conseguir dar las máximas vueltas posibles al circuito. Incluso antes de la llegada de las 3D, la cámara se situaba detrás del vehículo o dentro de la cabina. La técnica usada se basaba en simular la profundidad acercando cada elemento del circuito hacia la cámara por medio del escalado de gráficos. Al ser gráficos planos, se debía mantener en todo momento su orientación hacia la cámara. Después de *Pole Position*, aparecen otros títulos para PC-DOS bastante interesantes como *Grand Prix Circuit* (Accolade,Inc 1988) o *Indianápolis 500: The Simulation* (Papyrus Design/ EA, 1989). Con la llegada de Windows y de las DirectX se empiezan a desarrollar juegos más realistas con posibilidad multijugador y modos arcade o Grand Prix. Los años 90 fue una gran década de desarrollo para la F-1. La desarrolladora Papyrus Design arrasaba con sus títulos en el mercado con juegos como la serie *Indy Car Racing*.

El uso casi necesario de un dispositivo de juego como un gamepad o un volante hacía de estos juegos los mejores candidatos para las plataformas de consolas. Por ello, prácticamente la mayoría de títulos para PC eran versiones de ellas. Quizás la mejor de las primeras conversiones de juegos de PlayStation a PC sea *Formula 1* de Psygnosis 1996. El juego se desarrollaba completamente en 3D, además, se disfrutaba ya de hasta 17 circuitos reales, efectos climatológicos, coches dinámicos e incluso comentarista de la carrera. Un año después, Ubi Soft crea un precedente con *F1 Racing Simulation*, proporcionando un juego completamente en 3D con varias cámaras y corriendo a 640 x 480 de resolución. Además soportaba tanto gamepad y joystick como volante y pedales. Ubi suministró con este título la posibilidad de que jugasen por red has-



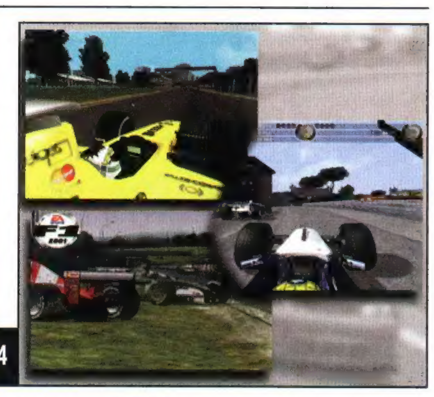
1
Dos clásicos de la fórmula 1: Pole Position (arriba) y Grand Prix Circuit (abajo)



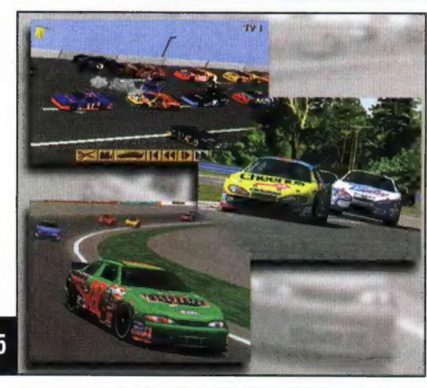
2
(Arriba) Indy Car Racing, de los primeros títulos de Papyrus Design. (Abajo) La mejor conversión para PC de los primeros juegos de PSX.



3
El modo multijugador no era ningún secreto para F1 Racing Simulator (arriba) y Monaco Grand Prix Racing Simulation 2 (abajo).


4

La serie F1 de EA Sports es lo mejor en simulación de la Fórmula 1.


5

La competición NASCAR fielmente representada a lo largo de la historia por la desarrolladora Papyrus Design.


6

Sega Rally 2 (arriba) y Colin McRae (abajo), los dos juegos de Rally más espectaculares y con mayor aceptación en el mercado.

ta 8 jugadores y hasta 2 por módem. Pero el que rompió moldes en el juego multijugador fue, sin dudas, Ubi Soft de nuevo, con *Monaco Grand Prix Racing Simulation 2* en 1999. Este estupendo título permitía multitud de modalidades multijugador, desde dos jugadores en el mismo ordenador hasta 8 por LAN o internet sin ningún tipo de problemas. Con la llegada del año 2000 estos simuladores adquieren un realismo increíble debido al avance en las tarjetas aceleradoras y por supuesto al desarrollo de dispositivos de juegos cada vez más complejos y precisos. La compañía EA Sports irrumpe en esta modalidad de juegos presentando en el 2000 *F-1 2000* y *F-1 Season 2000* con la licencia de *FIA* para simular el campeonato mundial. El gran éxito obtenido da pie a la publicación, un año después, del que quizás sea el juego de fórmula 1 que crea un punto de inflexión en la historia, *F-1 2001*. El realismo alcanzado es extremo, como detalle, decir que los mecánicos de los boxers se animaron por *motion capture*, sin contar con el realismo alcanzado en la conducción y la posibilidad de cambiar los parámetros del vehículo como cambios en los alerones o el freno, lo cual ayuda al jugador a ganar décimas de segundo en cada vuelta.

NASCAR Y RALLY

Si hay un simulador que destaque por su espectacularidad, estamos hablando de los juegos basados en los campeonatos NASCAR norteamericanos. Técnicamente, las técnicas utilizadas son iguales que los simuladores de F-1. Básicamente, se cambian los modelos de los vehículos y la posición del conductor. La desarrolladora Papyrus Design fue la primera en pasar al PC esta competición en el año 1994 con *NASCAR Racing* la cual continúa hoy día con *NASCAR Racing 2003*. Dada la naturaleza de esta carrera se ha provisto de un sistema de deformación de vehículos en tiempo real así como climatología y lo mejor de todo, la configuración total del vehículo.

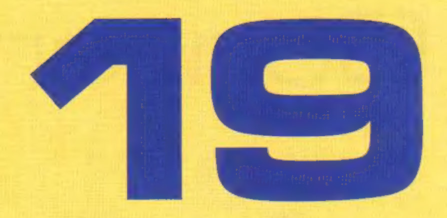
Tanto para *F-1* como *NASCAR* las carreras se realizan en circuitos cerrados, pero ¿qué ocurre cuando se quiere simular terreno natural? Evidentemente, la complejidad aumenta. El concepto de simuladores de Rally ya lo encontrábamos en las recreativas en los comienzos de los 80. Para PC se desarrollaron títulos un poco después como *African Raiders* (Tomahawk, 1986) o *4x4 OffRoad Racing* (Epix, Ltd 1988). Pero se han tenido que esperar algunos años para encontrar títulos más o menos decentes. Sin embargo, ya en 1993 podíamos disfrutar de un juego de Rally con posibilidad de cambios climáticos o conducir de noche o de día llamado *Rally* de *Europress Software*. Pero no fue hasta 1997 cuando se marcó un antes y un después en la simulación de Rallies con la aparición de una versión de consola llamada *Sega Rally ChampionShip* desarrollada por Sega. Poseía un motor gráfico 3D increíble que conseguía un rendimiento fuera de lo común en cualquier configuración. Además, estaba dotado de climatología, FMV (Full Motion Vídeo), voz y posibilidad multijugador. Posteriormente se lanza en el 2000 otra versión, también rompedora, centrado en el concepto arcade, sin ningún preámbulo antes de correr. Sin embargo, queda a la sombra del fantástico *Rally Master* de Infogrames que sale ese mismo año. Ya a partir del 2000, con las mejoras gráficas, aparecen títulos que aprovechan las posibilidades de las nuevas tecnologías; a los vehículos se les provee de cristal, cambios de morfología en el chasis por golpes o incluso manchas sufridas por el barro. Cada vez más, los simuladores de Rally van adquiriendo un mayor interés sobre todo gracias a la serie *Colin McRae* de Codemasters.



En el próximo número...

... terminaremos el repaso por los simuladores de velocidad.

Cuestionario Videojuegos



Preguntas

1. **¿Cómo podemos obtener la IP de un equipo local en notación 1.2.3.4 en Blitz3D?**
2. **¿Cómo podemos en Blitz3D iniciar un juego multijugador y crear un nuevo jugador sin utilizar la ventana de diálogo estándar de DirectPlay?**
3. **Escribe en Blitz3D una matriz de estructuras para definir 10 botones con doble estado gráfico para un menú.**
4. **¿Cómo podemos hacer en Blitz3D para elegir una opción del menú haciendo clic sobre un botón del mismo?**
5. **¿Dónde y cómo podemos crear un gradiente de color esférico con dos puntos de color en Paint Shop Pro?**
6. **¿Qué debemos tener en cuenta en el diseño de una textura para un sprite utilizado en un sistema de partículas?**
7. **En ModPlug Tracker hemos elegido el formato de canción de Impulse Tracker IT. ¿Cómo podemos añadir más canales a la canción?**
8. **¿Qué procedimientos debemos seguir en ModPlug para obtener nuevos sonidos y utilizarlos en los patrones?**
9. **¿Cómo podemos modificar el volumen de un clip de audio en Media Studio?**

Respuestas al cuestionario 18

▷ **1.** Protocolos TPC / IP y UDP

- TCP / IP se utiliza para la conexión por internet. La comunicación se realiza por envíos de paquetes ordenados y con un destino conocido. Sin embargo, son más lentos pero fiables. Ideal para juegos por turnos.
- UDP se suele utilizar para conectar ordenadores por red local. Es mucho más rápido pero no es fiable ya que los paquetes de información llegan desordenados. Ideal para juegos en tiempo real.

▷ **2.** TCP

```
Puerto_servidor= 60
Mensaje = OpenTCPStream ("127.0.0.2",servidor)
WriteString Mensaje, "Éste es el mensaje que envío"
CloseTCPStream Servidor
```

UDP

```
Puerto= 60
Mensaje = CreateUDPStream (Puerto)
WriteString Mensaje, "Mi nombre es Darío"
SendUDPMsg Mensaje, IP_Destino, Puerto_Destino
```

▷ **3.** Utilizando la instrucción "CameraProject" y "ProjectedX()":
Siendo el oyente la cámara, por ejemplo:

```
CameraProject (camara, EntityX (emisor), EntityY (emisor), EntityZ(emisor))
If ProjectedX#()>1 and ProjectedX#()>100 ChannelPan canal, valor panorámico
...
```

▷ **4.**

```
If Not ChannelPlaying(canal_musica)
    canal_musica=PlaySound(musica1)
EndIf
```

▷ **5.** Utilizando la librería de estilos de la herramienta *Preset Shapes*.

▷ **6.** Lo más importante a tener en cuenta es evitar que el jugador pierda la atención del juego y que aun así pueda ser visto con facilidad. Para ello, utilizamos formas estilizadas y concisas pero con colores vistosos.

▷ **7.** ModPlug trabaja en los siguientes formatos de canción: IT, MOD, S3M y XM.

▷ **8.** De izquierda a derecha: la nota musical, el sonido para esa nota, el volumen y el efecto.

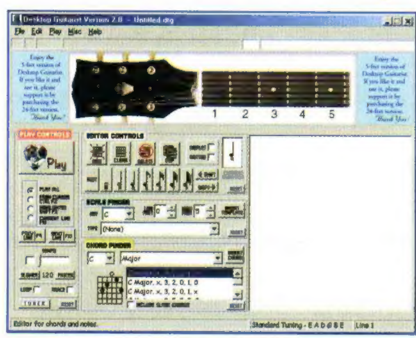
▷ **9.** Haciendo clic con el botón derecho sobre el clip aparecerá un menú flotante en el cual elegimos la opción "Video Filters". Aparecerá un menú de diálogo con la lista de filtros, los cuales podemos ir añadiendo.

▷ **10.** En la sección "Motion Path" de la ventana de edición del filtro.

Contenido CD-ROM 19

▶ AUDIO

■ Ace High MP3 Recorder
Graba tus archivos de audio en formato MP3 usando esta útil aplicación.

■ Desktop Guitarist 2


Los amantes de la música de guitarra podrán crear sus propias composiciones fácilmente.

■ Diet MP3
Comprime tus MP3 hasta el punto que tú desees gracias a esta utilidad.

■ AudioLabel CD Labeler
Crea tus propias etiquetas para los cds de audio.

■ Audio MP3 Maker
Convertidor de cds de audio a MP3, fácil de usar y rápido en sus resultados.

■ Piolet
Comparte tus archivos de música con otros gracias a este sencillo programa peer to peer.

▶ DISEÑO 2D

■ PhotoGraffiti


Crea divertidas imágenes gracias a esta simpática aplicación que te encantará.

■ UltraScreen Capture
Captura cualquier parte de la pantalla fácil y cómodamente.

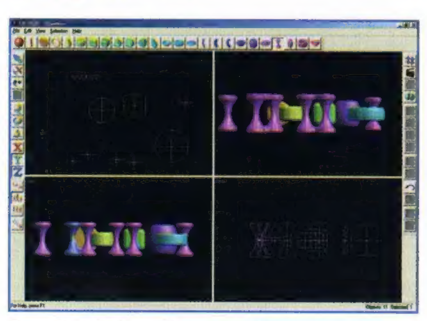
■ RasterVect
Crea y edita tanto imágenes vectoriales como mapas de bits con este programa.

■ PhotoBuilder
Mezcla distintas imágenes entre sí para crear fabulosas composiciones.

▶ DISEÑO 3D

■ D4Sound 2.5
Pequeño programa para crear figuras en tres dimensiones y mezclarlas además con música.

■ Imagine
Editor de entornos 3D muy sencillo de manejar.

■ Organica 1.0


Crea fantásticas figuras en 3D en pocos minutos gracias a este estupendo software.

■ Particle Illusion
Añade efectos de partículas y texturas en tus animaciones usando este programa.

▶ PROGRAMACIÓN

■ MojoWorld Generator 2.0
Crea tus propios mundos virtuales. Deberás registrarte en internet para conseguir una contraseña gratuita.

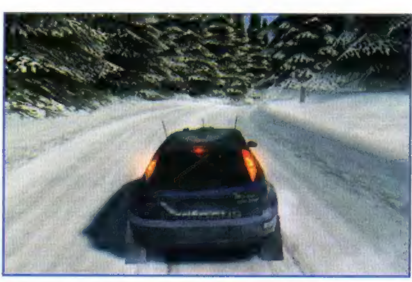
■ Rapid PLUS Express


Crea modelos gráficos interactivos de tus futuros productos.

■ HelpTron 2.3
Crea completas ayudas en HTML para tus juegos, compatibles con todos los Windows.

■ Solid Concept 3.0
Software para modelar en tres dimensiones, totalmente operativo durante un mes obteniendo el código gratuito en la web.

▶ JUEGOS

■ Colin McRae Rally


Corre a todo trapo e intenta ganar la carrera con este estupendo simulador de coches.

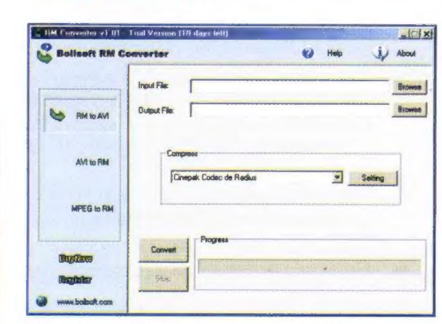
■ Nascar Racing 2002
Otro magnífico simulador de coches de carreras, con el que pasarás estupendos momentos.

■ SegaRally Championchip
Experimenta virtualmente la emoción de correr en una carrera de coches.

■ Zone of Fighters
Como siempre, nuestro juego.

▶ VÍDEO

■ Boomer Video
Estupenda aplicación para crear vídeo y audio en streaming fácilmente.

■ RM Converter


Convierte vídeos en formato RealMedia a otros múltiples formatos.

▶ EXTRAS
En este apartado encontrarás todos los ejemplos de los que hablamos en el coleccionable, para que no pierdas detalle.